Combinatorial optimization in Image Understanding: Graph cut-based image segmentation

김창익 교수

EE, KAIST

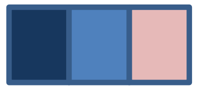
Outline

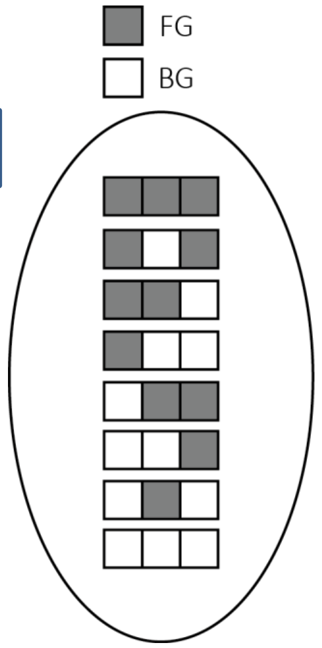
- Introduction to combinatorial optimization
- Energy-based image segmentation (object extraction)
- Graph cut-based image segmentation

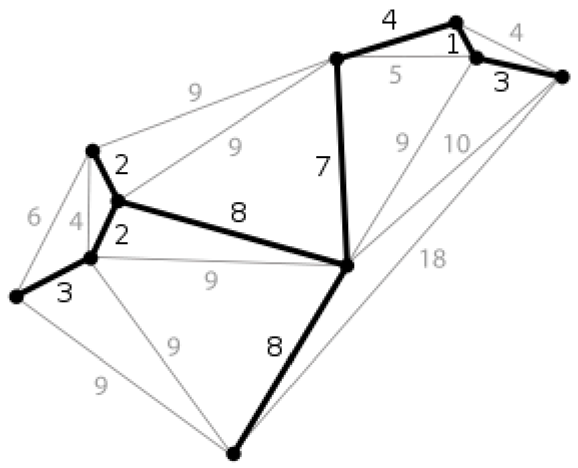
- Object Segmentation via Graph Cuts
- Automatic extraction of objects
- Normalized Cut

Introduction

- Combinatorial optimization problems
  - In combinatorial optimization, the set of possible solutions S has a finite number of elements.
  - Any combinatorial optimization problem is also a discrete problem.
  - A solution can be expressed as a combination of representation of data. (i.e., it is finding an optimal object from a finite set of objects)

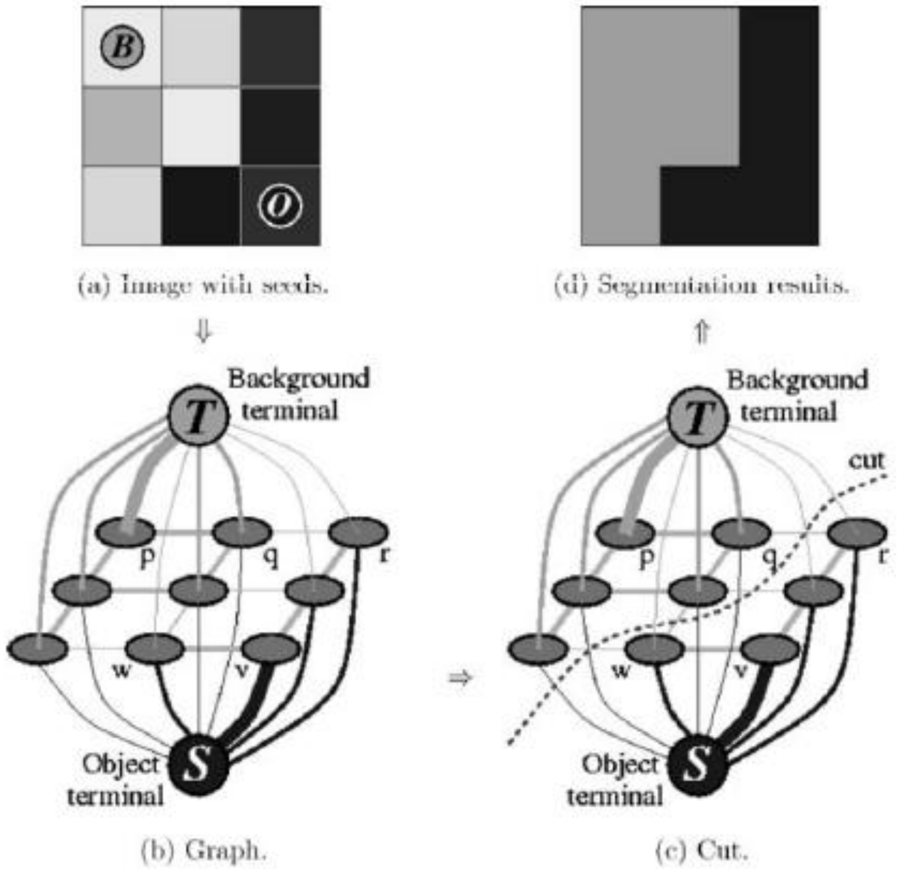






Introduction

- Object Segmentation (pixel labeling) via Graph Cuts



(a) Image with seeds.

(d) Segmentation results.

(b) Graph.

(c) Cut.

Energy based Image Segmentation

- Energy-based segmentation methods can be distinguished by the type of energy function they use and by the optimization technique for minimizing it. The majority of standard algorithms can be divided into two large groups:
  - (A) Optimization of a functional defined on a continuous contour or surface.
  - (B) Optimization of a cost function defined on a discrete set of variables.

- The standard methods in group (A) formulate segmentation problem in the domain of continuous functions R^{∞}.
- Most of them rely on a variational approach and gradient decent for optimization.
- The corresponding numerical techniques are based on finite differences or on finite elements.

Energy based Image Segmentation

- The segmentation methods in group (B) either directly formulate the problem as combinatorial optimization in finite dimensional space Z^n or optimize some discrete energy function whose minima approximates solution of some continuous problem.

Energy based Image Segmentation

	Variational methods (optimization in R^∞)	Combinatorial methods (optimization in Z^n)
Explicit boundary representation	Snake algorithm (see next slide)	Dynamic programming
Implicit boundary representation	Level-sets (see next slide)	Graph cuts

Energy based Image Segmentation

- Ex) Snake Algorithm
  - The problem at hand is to find a contour v(s) that minimize the energy functional

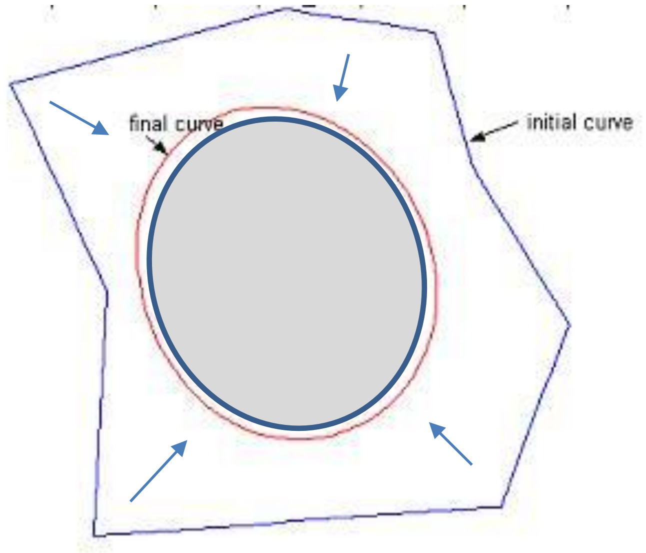
$$E^*_{snake} = \int_s \frac{1}{2}(\alpha(s)\,|v_s|^2 + \beta(s)\,|v_{ss}|^2) + E_{image}(v(s))ds$$

$$E^*_{snake} = \int_0^1 E_{snake}(s, \mathbf{v}(s), \mathbf{v}'(s), \mathbf{v}''(s))ds.$$

  - Using variational calculus and by applying Euler-Lagrange differential equation we get following equation

$$\frac{d^2}{ds^2}E_{\mathbf{v}_{ss}} - \frac{d}{ds}E_{\mathbf{v}_s} + E_{\mathbf{v}} = 0.$$

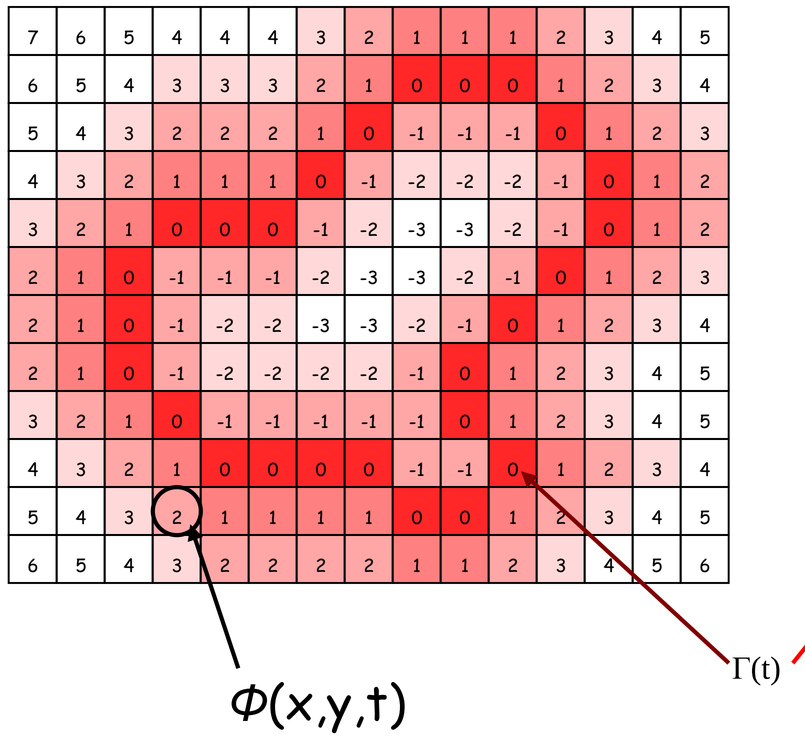
$$-\frac{d}{ds}\left(\alpha(s)\frac{d\mathbf{v}}{ds}\right) + \frac{d^2}{ds^2}\left(\beta(s)\frac{d^2\mathbf{v}}{ds^2}\right) + \nabla E_{img}(\mathbf{v}(s)) = 0.$$



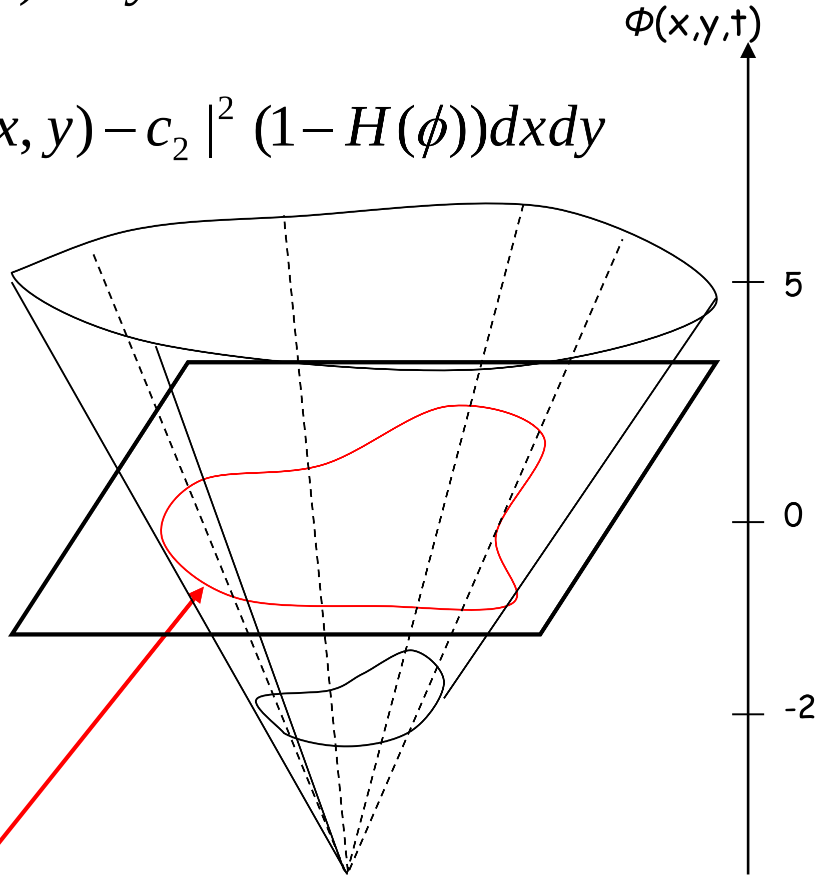
Energy based Image Segmentation

- Ex) Level sets

$$F(c_1,c_2,\phi)=\mu\int_\Omega|\nabla H(\phi)|\,dxdy+\nu\int_\Omega H(\phi)dxdy$$
$$+\lambda\int_\Omega|u_0(x,y)-c_1|^2\,H(\phi)dxdy+\lambda\int_\Omega|u_0(x,y)-c_2|^2\,(1-H(\phi))dxdy$$



7	6	5	4	4	4	3	2	1	1	1	2	3	4	5
6	5	4	3	3	3	2	1	0	0	0	1	2	3	4
5	4	3	2	2	2	1	0	-1	-1	-1	0	1	2	3
4	3	2	1	1	1	0	-1	-2	-2	-2	-1	0	1	2
3	2	1	0	0	0	-1	-2	-3	-3	-2	-1	0	1	2
2	1	0	-1	-1	-1	-2	-3	-3	-2	-1	0	1	2	3
2	1	0	-1	-2	-2	-3	-3	-2	-1	0	1	2	3	4
2	1	0	-1	-2	-2	-2	-2	-1	0	1	2	3	4	5
3	2	1	0	-1	-1	-1	-1	-1	0	1	2	3	4	5
4	3	2	1	0	0	0	0	-1	-1	0	1	2	3	4
5	4	3	2	1	1	1	1	0	0	1	2	3	4	5
6	5	4	3	2	2	2	2	1	1	2	3	4	5	6



Graph Cut based Segmentation

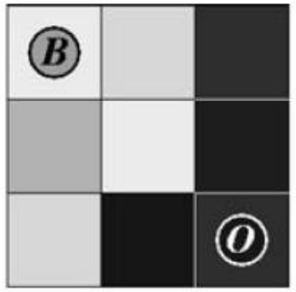
- Why Graph Cuts?
  - All previous combinatorial methods for object segmentation use discrete variables whose values encode "direction" of a path along the graph.
  - Many path-based methods use Dynamic Programming (DP) to compute optimal paths.
  - In contrast to the earlier "path-based" combinatorial methods, this can be seen as a "region-based" approach to encoding image segments.

Graph Cut based Segmentation

- Why Graph Cuts? (cont'd)
  - Robust and efficient optimization in N-D.
    - We start with a discrete energy formulation and directly solve it with an exact graph-based optimization method.
    - In contrast, variational methods inherently rely on approximating numerical schemes (e.g. finite differences or finite elements) that must be very carefully designed to insure robustness.
    - Segmentation results generated by two variational techniques using the same energy may depend on their implementation details.

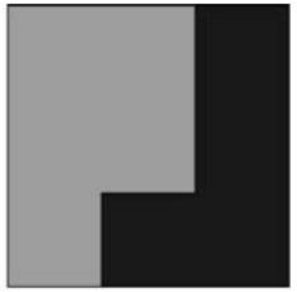
Object Segmentation via Graph Cuts

Object Segmentation via Graph Cuts



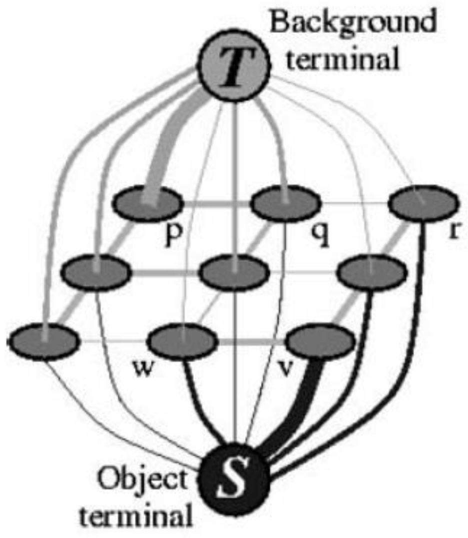
(a) Image with seeds.

⇓



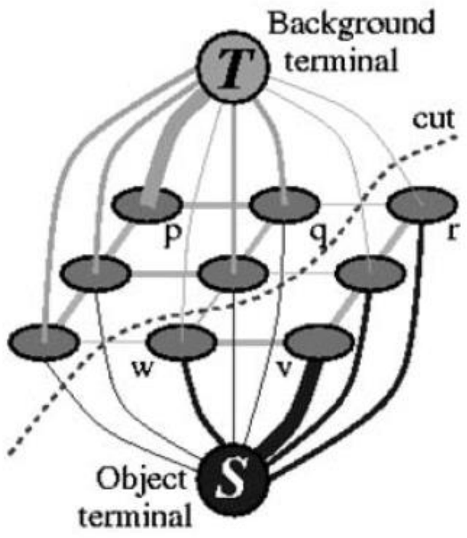
(d) Segmentation results.

⇑



(b) Graph.

⇒



(c) Cut.

Object Segmentation via Graph Cuts

- Basic Idea
  - A graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ is defined as a set of nodes or vertices V and a set of edges $\mathcal{E}$ connecting neighboring nodes.
  - An s-t cut is a subset of edges $C \subset \mathcal{E}$ such that the terminals S and T become completely separated on the induced graph $\mathcal{G}(C) = \langle \mathcal{V}, \mathcal{E} \backslash C \rangle$.

Object Segmentation via Graph Cuts

- Segmentation Energy
  - *P* is a set of pixels (or voxels).
  - Neighborhood system represented by a set of N.
  - Let $\boldsymbol{A} = (A_1, \ldots, A_p, \ldots A_{|P|})$ be a binary vector whose components $\boldsymbol{A}_p$ can be either “obj” or “bkg”.
  - Vector $\boldsymbol{A}$ defines a segmentation.

Object Segmentation via Graph Cuts

- Cost function

$$E(A) = \lambda \cdot R(A) + B(A) \qquad (1)$$

where

$$R(A) = \sum_{p \in \mathcal{P}} R_p(A_p) \quad \textit{(regional term)}$$

$$B(A) = \sum_{\{p,q\} \in \mathcal{N}} B_{p,q} \cdot \delta_{A_p \neq A_q} \quad \textit{(boundary term)}$$

$B_{p,q}$ is large when I_p and I_q are similar.

and

$$\delta_{A_p \neq A_q} = \begin{cases} 1 & \text{if } A_p \neq A_q \\ 0 & \text{if } A_p = A_q. \end{cases}$$

Object Segmentation via Graph Cuts

- Assigning Costs
  - Regional cost

$$R_p(\text{"obj"}) = -\ln \Pr(I_p|\text{"obj"})$$
$$R_p(\text{"bkg"}) = -\ln \Pr(I_p|\text{"bkg"})$$

-ln 0.1 = 2.3
-ln 0.5 = 0.7
-ln 1 = 0

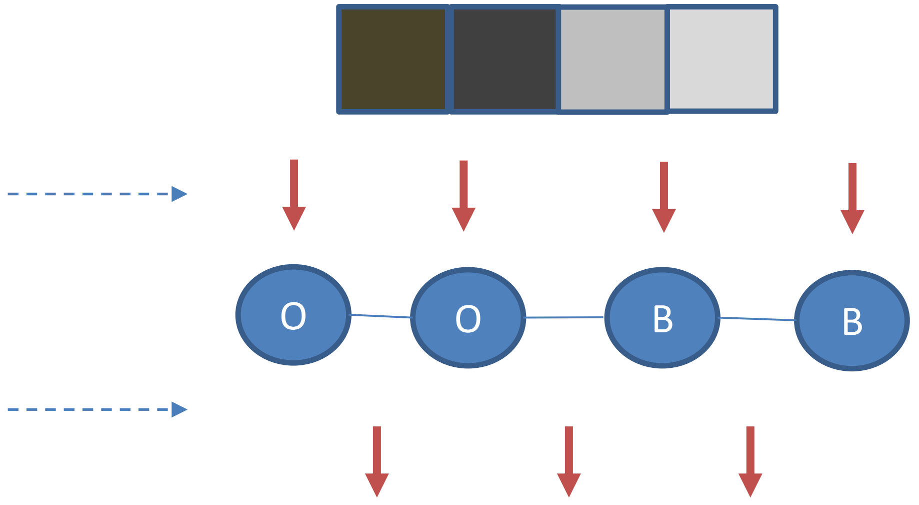
  - Boundary cost
    - $B_{p,q}$ is large when pixels *p* and *q* are similar in their intensity.
    - $B_{p,q}$ is close to zero when the two are very different.

$$B_{p,q} \propto \exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2}\right) \cdot \frac{1}{dist(p,q)}$$

Object Segmentation via Graph Cuts

- Assigning Costs
  - Regional cost

$$R_p(\text{"obj"}) = -\ln \Pr(I_p | \text{"obj"})$$
$$R_p(\text{"bkg"}) = -\ln \Pr(I_p | \text{"bkg"})$$



  - Boundary cost
    - $B_{p,q}$ is large when pixels *p* and *q* are similar.
    - $B_{p,q}$ is close to zero when very different.

$$B_{p,q} \propto \exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2}\right) \cdot \frac{1}{dist(p,q)}$$

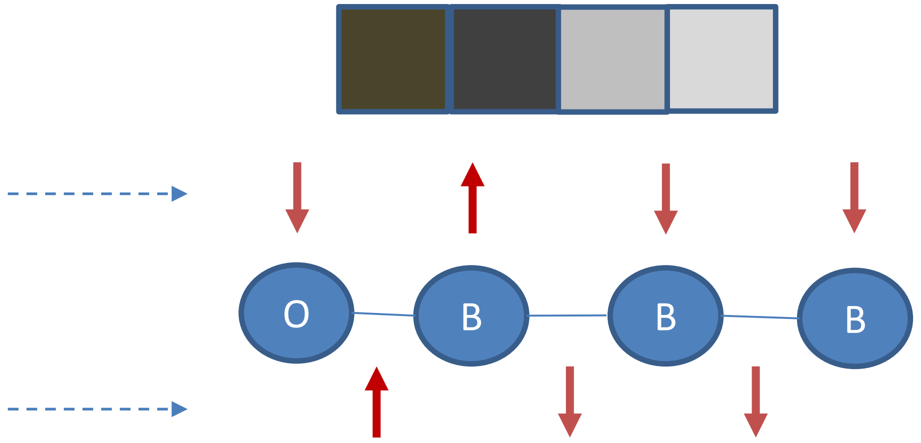
-ln 0.1 = 2.3
-ln 0.5 = 0.7
-ln 1 = 0

$$\delta_{A_p \neq A_q} = \begin{cases} 1 & \text{if } A_p \neq A_q \\ 0 & \text{if } A_p = A_q. \end{cases}$$

Object Segmentation via Graph Cuts

- Assigning Costs
  - Regional cost

$$R_p(\text{“obj”}) = -\ln \Pr(I_p|\text{“obj”})$$
$$R_p(\text{“bkg”}) = -\ln \Pr(I_p|\text{“bkg”})$$



  - Boundary cost
    - $B_{p,q}$ is large when pixels p and q are similar.
    - $B_{p,q}$ is close to zero when very different.

$$B_{p,q} \propto \exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2}\right) \cdot \frac{1}{dist(p,q)}$$

-ln 0.1 = 2.3
-ln 0.5 = 0.7
-ln 1 = 0

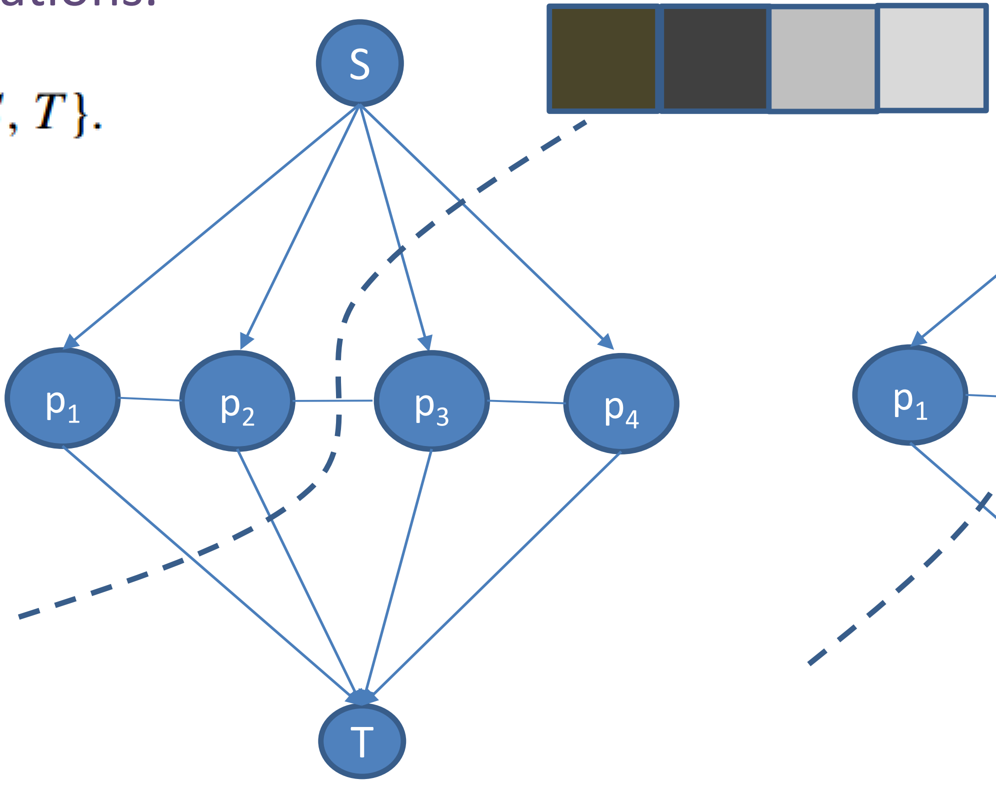
$$\delta_{A_p \neq A_q} = \begin{cases} 1 & \text{if } A_p \neq A_q \\ 0 & \text{if } A_p = A_q. \end{cases}$$

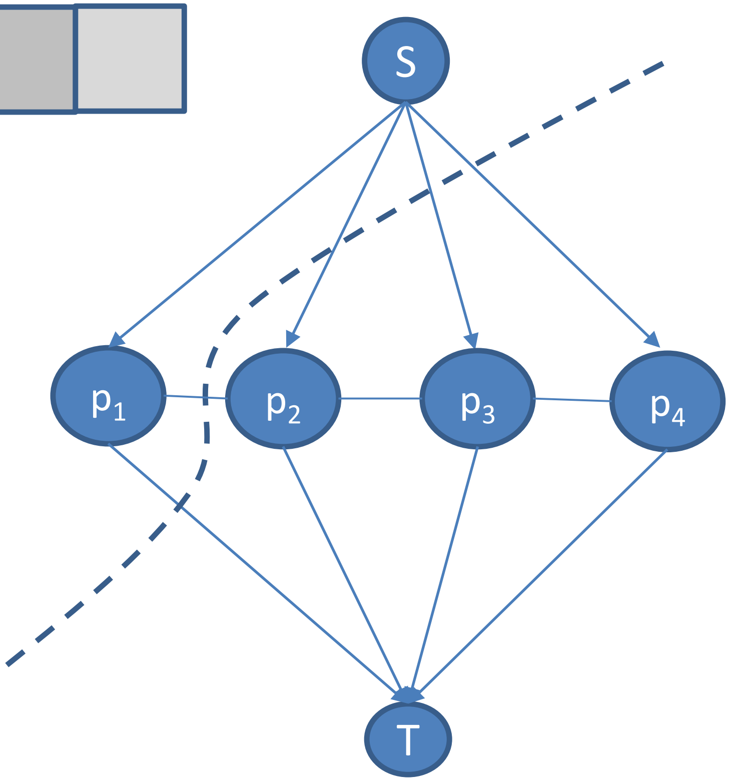
Object Segmentation via Graph Cuts

- Optimal solution via Graph Cuts
  - Theorem: The minimum cut of $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ minimizes the cost function (1) among all segmentations.

$\mathcal{V} = \mathcal{P} \cup \{S, T\}.$

$\mathcal{E}$: see the table following





Cost low

Cost high

Y. Boykov and G. Funka-Lea, "Graph Cuts and Efficient N-D Image Segmentation," IJCV 70(2), 2006.

Object Segmentation via Graph Cuts

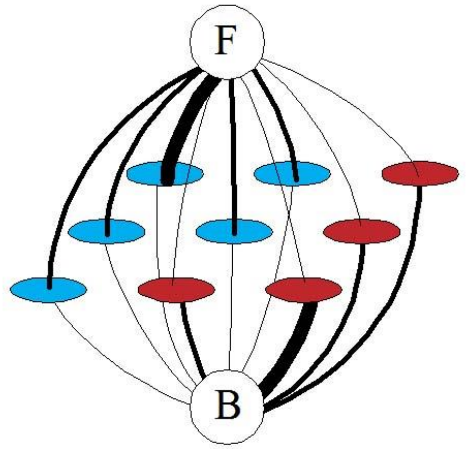
- Max-flow/Min-cut theorem
  - The maximum amount of flow is equal to the capacity of a minimum cut
  - Min Cut can be made by using Ford-Fulkerson Algorithm to find t he max-flow in the graph.

Object Segmentation via Graph Cuts

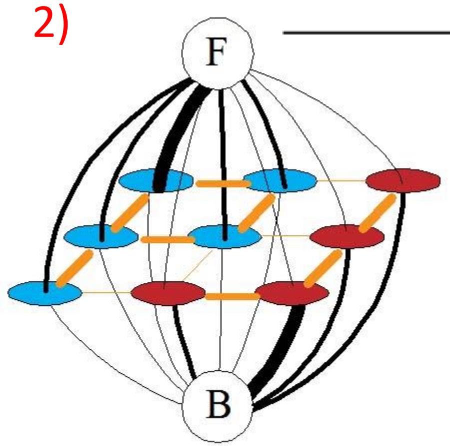
- Cost (weight) terms

Graph arc	Cost	
(p, q)	$B_{(p,q)}$	$for (p, q) \in N$
(s, p)	$\lambda R_p(bgd)$	$for\ p \in I, p \notin (O \cup B)$
	K	$for\ p \in O$
	0	$for\ p \in B$
(p, t)	$\lambda R_p(obj)$	$for\ p \in I, p \notin (O \cup B)$
	0	$for\ p \in O$
	K	$for\ p \in B$

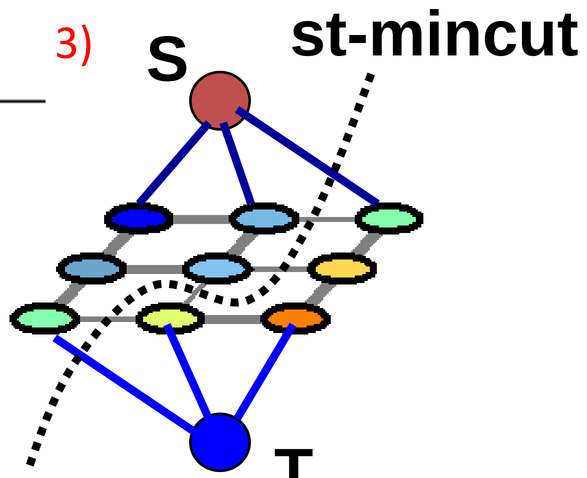
1)



2)



3)



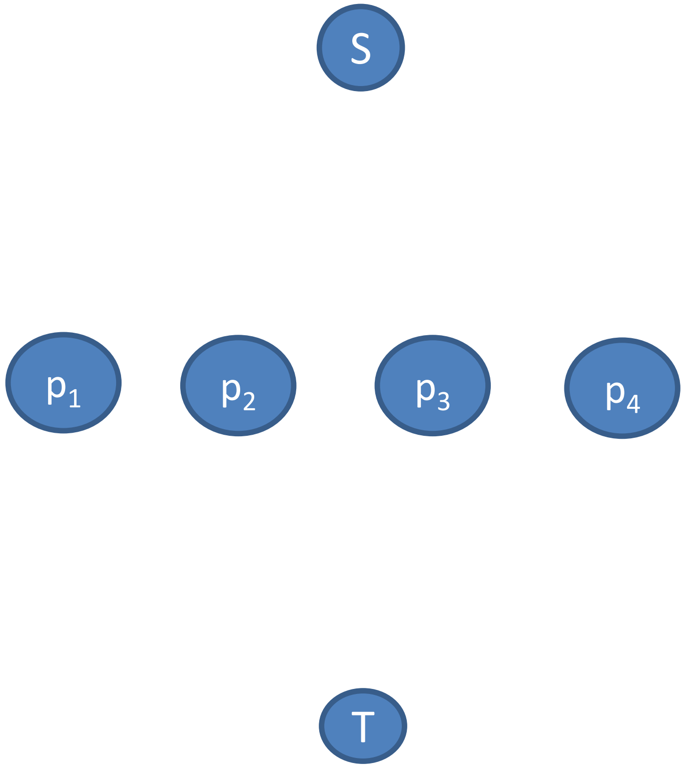
Object Segmentation via Graph Cuts

- Algo. Graph cut segmentation
  1. Create an arc-weighted directed graph corresponding in size and dimensionality to the image to be segmented.
  2. Identify object and background seeds-example points required to be part of the background or object(s) in the final segmentation. Create two special graph nodes-source *s* and sink *t*; connect all seeds with either the source or the sink node based on their object or background label.
  3. Associate appropriate arc cost with each link of the formed graph according to the table in the previous slide.
  4. Use one of the available maximum flow graph optimization algorithms to determine the graph cut.
  5. The minimum s-t cut solution identifies the graph nodes that correspond to the image boundaries separating the object(s) and the background.

Object Segmentation via Graph Cuts

- Implementation
  - Codes for GC

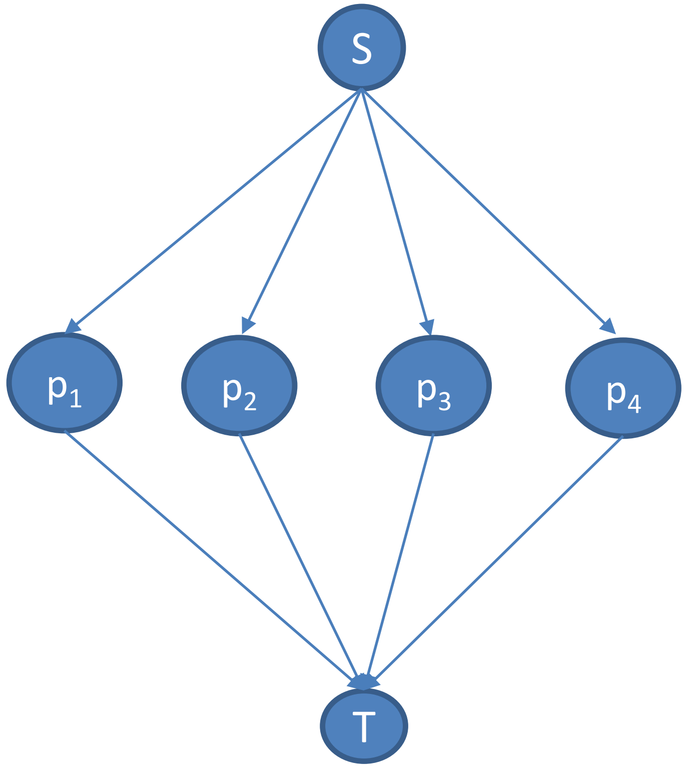
```
Graph *g;

For all pixels p

    /* Add a node to the graph */
    nodeID(p) = g->add_node();

    /* Set cost of terminal edges */
    set_weights(nodeID(p), fgCost(p), bgCost(p));

end

for all adjacent pixels p,q
    add_weights(nodeID(p), nodeID(q),  cost);
end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)
```



Object Segmentation via Graph Cuts

- Implementation
  - Codes for GC

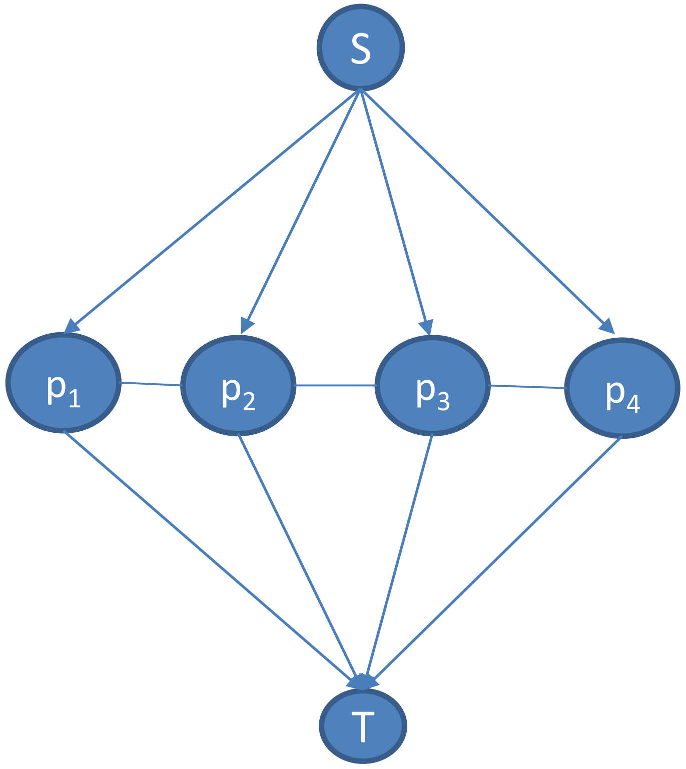
```
Graph *g;

For all pixels p

        /* Add a node to the graph */
        nodeID(p) = g->add_node();

        /* Set cost of terminal edges */
        set_weights(nodeID(p), fgCost(p), bgCost(p));

end

for all adjacent pixels p,q
        add_weights(nodeID(p), nodeID(q),  cost);
end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)
```



Object Segmentation via Graph Cuts

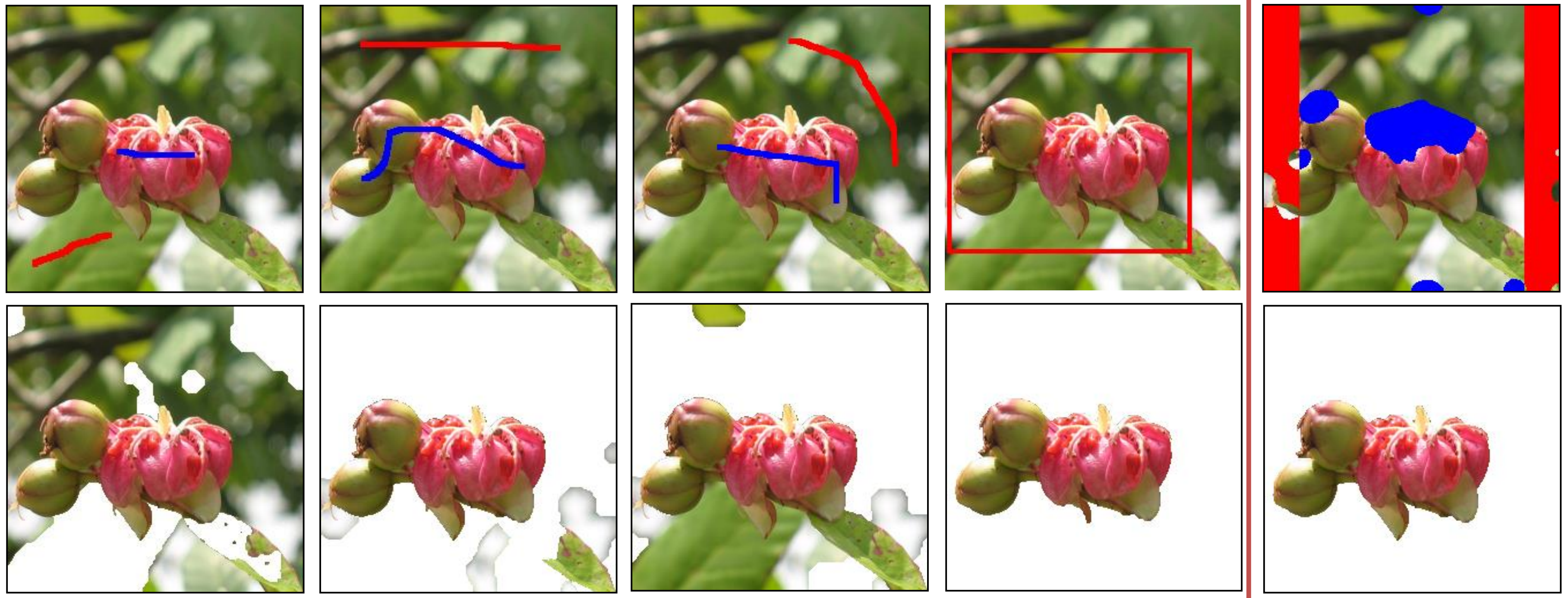
- Implementation
  - Codes for GC

```
Graph *g;

For all pixels p

        /* Add a node to the graph */
        nodeID(p) = g->add_node();

        /* Set cost of terminal edges */
        set_weights(nodeID(p), fgCost(p), bgCost(p));

end

for all adjacent pixels p,q
        add_weights(nodeID(p), nodeID(q),  cost);
end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)
```



Object Segmentation via Graph Cuts

- Examples of graph cut segmentation
  - ❖ Segmentation results heavily depend on user interaction.



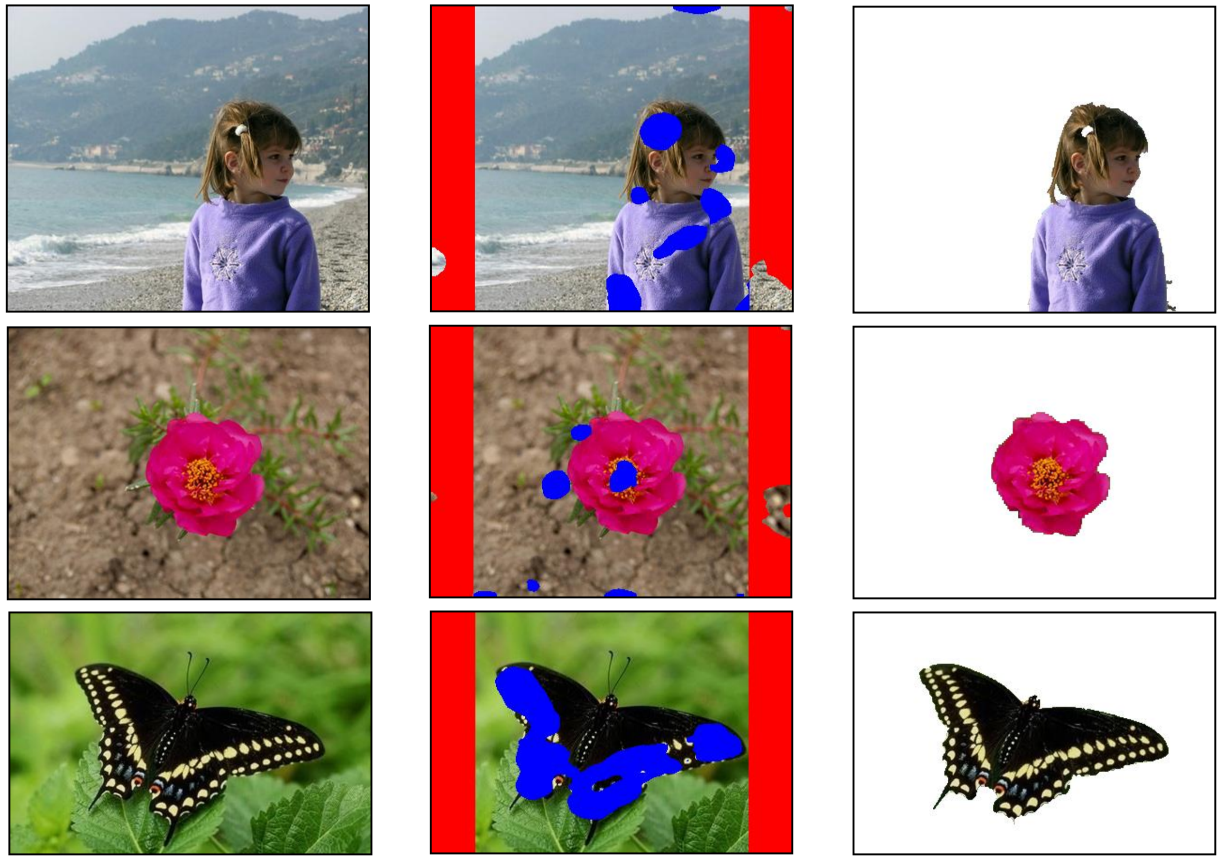
Interactive Graph Cuts [1] | Interactive Graph Cuts [1] | Interactive Graph Cuts [1] | GrabCut [2] | Markov Chain Embedded Graph Cuts (proposed)

[1] Y. Boykov and G. Funka-Lea, "Graph cuts and efficient N-D image segmentation," *International Journal of Computer Vision*, vol. 70, no. 2, pp. 109-131, 2006.

[2] C. Rother, V. Kolmogorov, and A. Blake, " "GrabCut" - Interactive Foreground Extraction using Iterated Graph Cuts," *ACM Transactions on Graphics*, pp. 309-314, 2004.

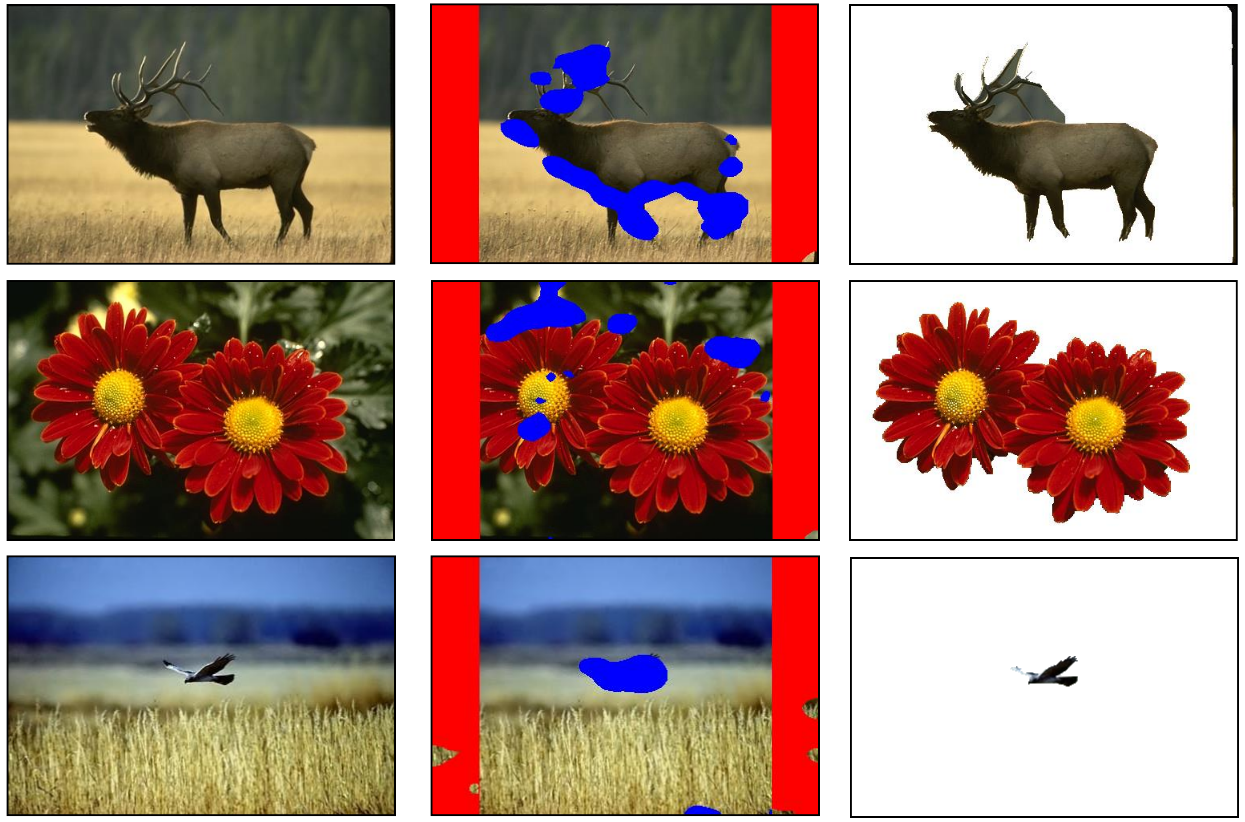
[**proposed**] C. Jung, B.Kim, and C.Kim, "Automatic segmentation of salient objects using iterative reversible graph cut," in *Proc. of IEEE International Conference on Multimedia & Expo (ICME)*, pp. 590-595, Singapore, July 2010.

Proposed Automatic Segmentation



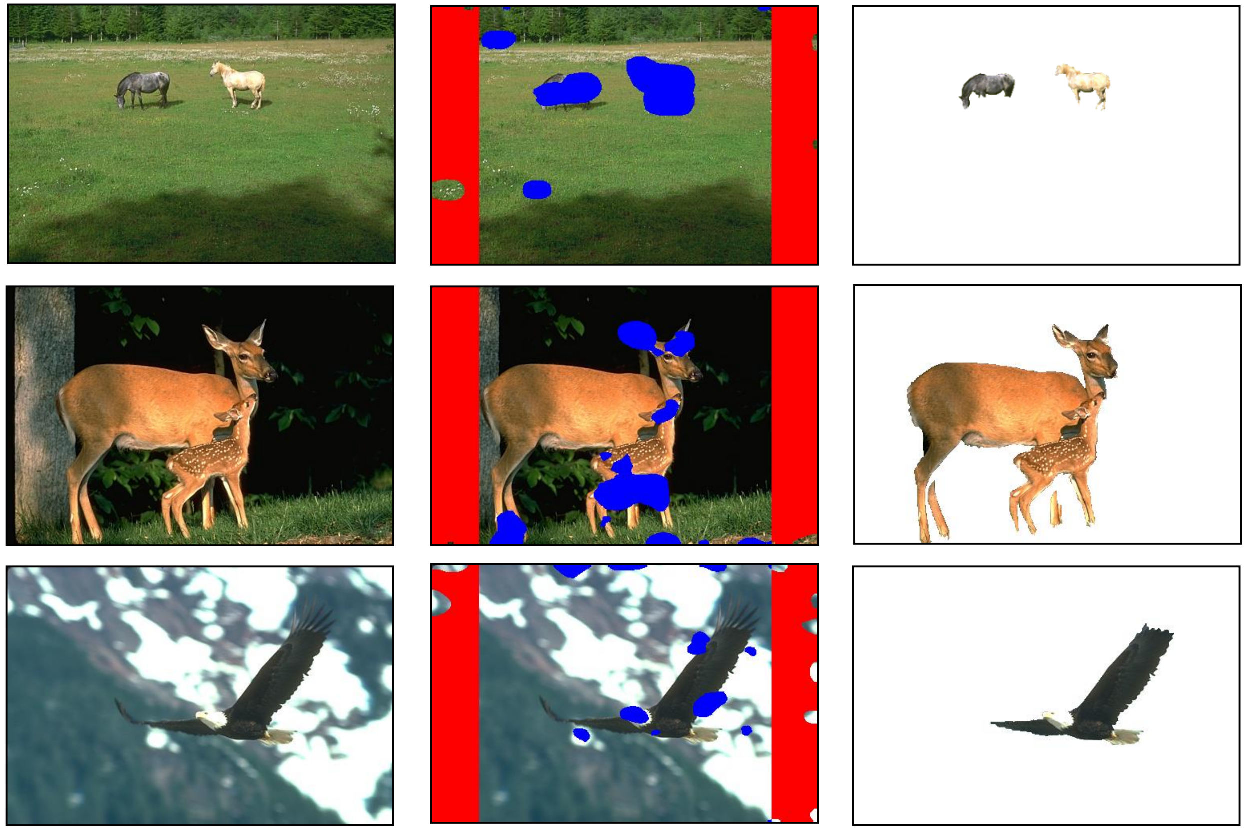
[**proposed**] C. Jung, B.Kim, and C.Kim, "Automatic segmentation of salient objects using iterative reversible graph cut," in *Proc. of IEEE International Conference on Multimedia & Expo (ICME)*, pp. 590-595, Singapore, July 2010.

Proposed Automatic Segmentation



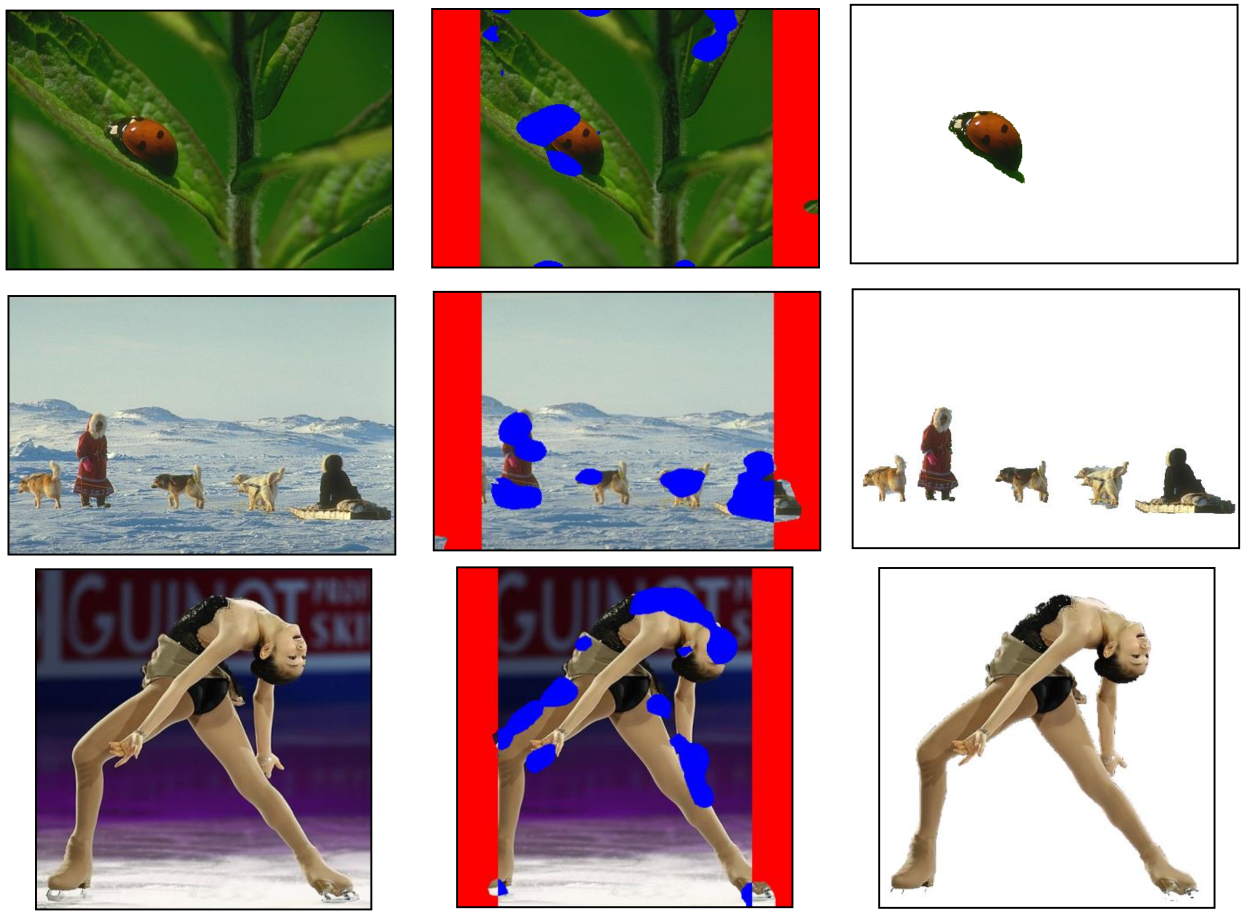
Input Image **Extracted Seed** **Proposed Algorithm**

Proposed Automatic Segmentation



Input Image Extracted Seed Proposed Algorithm

Proposed Automatic Segmentation



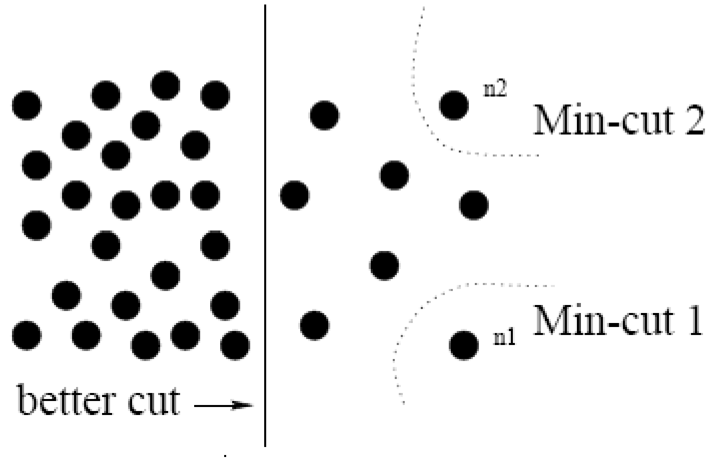
Input Image

Extracted Seed

Proposed Algorithm

Normalized Cut

- Minimum cut criteria favors cutting small sets of isolated nodes in the graph



- Normalized cut criterion can be computed efficiently by solving a generalized eigenvalue problem.

References

- Y. Boykov and G. Funka-Lea, “Graph Cuts and Efficient N-D Image Segmentation,” IJCV 70(2), 109-131, 2006.
- Image processing, analysis, and machine vision, Sonka et al., Thomson, 2008.
- J. Shi and J. Malik, “Normalized Cuts and Image Segmentation,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pp. 888-905, Aug. 2000.